

**IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
AUSTIN DIVISION**

K.MIZRA LLC,

Plaintiff,

v.

GOOGLE LLC,

Defendant.

Civil Action No.: 1:26-cv-00754

Jury Trial Demanded

COMPLAINT FOR PATENT INFRINGEMENT

Plaintiff K.Mizra LLC ("K.Mizra") files this Complaint for Patent Infringement against Defendant Google LLC ("Google"), alleging as follows:

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
II.	THE PARTIES.....	2
III.	JURISDICTION AND VENUE.....	2
IV.	GENERAL ALLEGATIONS	3
A.	The '120 Patent.....	3
1.	Prior Licensing and Litigation Concerning the '120 Patent	3
2.	Problems in 2007 Solved By the '120 Patent	4
3.	K.Mizra's Asserted '120 Patent Claims Are Presumed Valid and Patent-Eligible	8
4.	Google's Infringing Vertex AI Studio Product	10
B.	The '717 Patent.....	11
1.	Prior Licensing and Litigation Concerning the '717 Patent	12
2.	Problems in 2007 Solved By the '717 Patent	12
3.	K.Mizra's Asserted '717 Patent Claims Are Presumed Valid and Patent-Eligible	13
4.	Google's Infringing Nest Devices.....	19
C.	Google's Knowledge of the Asserted Patents.....	23
V.	FIRST CLAIM FOR RELIEF (Patent Infringement Under 35 U.S.C. § 271 of the '120 Patent)	24
VI.	SECOND CLAIM FOR RELIEF (Patent Infringement Under 35 U.S.C. § 271 of the '717 Patent)	33
VII.	REQUEST FOR RELIEF	48
VIII.	DEMAND FOR JURY TRIAL.....	49

I. INTRODUCTION

1. K.Mizra is a patent licensing company run by experienced management. The company focuses on high-value, high-quality patents and owns patent portfolios originating from a wide array of inventors, including patents and patent portfolios developed by well-known multinational corporations such as IBM, Intel, Rambus and others, as well as from research institutes such as Nederlandse Organisatie voor Toegespast Natuurwetenschappelijk Onderzoek (Netherlands Organization for Applied Scientific Research). By focusing on high-quality patents, K.Mizra provides a secondary market that enables inventors to recoup their research and development investments and to continue their innovations. K.Mizra offers licenses to its patents on reasonable terms and, in this way, plays an important role in the development of technologies that improve businesses and lives.

2. K.Mizra is the owner by assignment of United States Patent No. 8,438,120 ("the '120 Patent"). The '120 Patent was invented by Stephan Alexander Raaijmakers of the Nederlandse Organisatie Voor Toegepast-Natuurwetenschappelijk Onderzoek TNO, a well-known and widely respected technology research institution ("TNO").

3. K.Mizra is also the owner by assignment of United States Patent No. 8,144,717 ("the '717 Patent" or "the Mesh Networking Patent"). The '717 Patent was invented by Marinus Johannes Blange and Miodrag Djurica of the TNO.

4. The '120 Patent and '717 Patent are collectively referred herein as "the Asserted Patents." K.Mizra remains ready, willing, and able to provide commercially reasonable licenses for its various patented technologies, including its Asserted Patents, to all entities who wish or need to use its covered technologies internally or in connection with products or services offered to others. As outlined below, Google is one such entity.

II. THE PARTIES

5. K.Mizra is a Delaware limited liability company with a mailing address of 777 Brickell Avenue, #500-96031, Miami, Florida 33131, and operates in Florida.

6. K.Mizra is the owner by assignment of the Asserted Patents.

7. Upon information and belief, Google is a corporation organized and existing under the laws of the state of Delaware, with a principal place of business at 1600 Amphitheatre Parkway, Mountain View, California 94043.

8. Upon information and belief, Google maintains an office in this District at 500 West Second Street, Austin, Texas 78701.

9. Upon information and belief, Google employs hundreds of people in its Austin, Texas office, including persons with knowledge of the Google products and technology at issue in this case.

10. Upon information and belief, Google is also registered to do business in Texas and may be served with process by serving Corporation Service Company, Google's registered agent in Texas located at 211 East Seventh Street, Suite 620, Austin, Texas 78701-3218.

III. JURISDICTION AND VENUE

11. This is an action for patent infringement under the patent laws of the United States, 35 U.S.C. §§ 1 *et seq.*, including 35 U.S.C. §§ 271, 281, and 284, among others. The Court has subject-matter jurisdiction over the claims raised in this action pursuant to 28 U.S.C. §§ 1331 and 1338(a).

12. This Court has personal jurisdiction over Google by virtue of, *inter alia*, its conduct of business in this District, its registration to do business in Texas, its appointment of a registered agent in Texas, its employment of hundreds of persons in its Austin office located in this District,

and its substantial, continuous, and systematic contacts with the state of Texas and this District. Google intentionally markets and sells its infringing products directly and through agents to residents of Texas, enjoys substantial income from its business activities in the state of Texas and this District, and/or directly, by its own actions, and/or in combination with actions of customers and others under its control, has committed acts of patent infringement in this District at least by selling infringing products in this District.

IV. GENERAL ALLEGATIONS

A. The '120 Patent

13. K.Mizra is the sole owner by assignment of the '120 Patent with the full and exclusive right to bring suit to enforce the Patent. K.Mizra is also entitled to sue to collect damages for all past infringement of the '120 Patent.

14. The '120 Patent, titled "Machine Learning Hyperparameter Estimation," was issued by the United States Patent and Trademark Office ("USPTO") to Inventor Stephen Alexander Raaijmakers on May 7, 2013. A true and correct copy of the '120 Patent is attached hereto as Exhibit 1 and incorporated by reference.

15. The '120 Patent is a U.S. National Phase filing of PCT/NL2008/050247 and claims priority to European Application Nos. 07106963.7 filed April 25, 2007 and 07112037.2 filed July 9, 2007.

1. Prior Licensing and Litigation Concerning the '120 Patent

16. The '120 Patent has previously been licensed to various industry participants.

17. Google is not and has never been a licensee of the '120 Patent nor has it had nor does it have any rights to use technologies covered by the Patent. Google thus has no ownership or other rights (and is entitled to no rights) relating to the '120 Patent.

18. No party licensed to the '120 Patent has used the technology claimed by the '120 Patent in a manner that requires marking under 35 U.S.C. § 287. K.Mizra is therefore not barred from seeking past damages for infringement of the '120 Patent under 35 U.S.C. § 287(a).

2. Problems in 2007 Solved By the '120 Patent

19. The technology described in the '120 Patent was invented by Stephen Raaijmakers and generally relates to improving the operation of Artificial Intelligence ("AI") through the improved automatic, systematic and computer implemented training techniques. More specifically, AI typically uses models to perform specific tasks, and, at its most basic level, an AI model is a program that has been "trained" on a set of data to recognize certain patterns or to make certain decisions without further human intervention. For example, an AI model could be used to automatically detect spam or other unwanted emails in a user's email inbox or distinguish between ripe and unripe fruit to be picked, etc. The '120 Patent uses the term "classifier" to refer to these types of AI models, as they are used to separate inputs (emails or fruit) into different classifications (spam or not spam/ripe or unripe).

20. AI models use variables called "parameters" to determine what result it should produce (e.g., to determine whether to classify an email as spam or not spam). These parameters are not set manually. Instead, they are learned automatically during the training process. During training, the model may be exposed to a labeled dataset and the parameters can be adjusted to minimize prediction errors. For instance, in the spam email example, the AI model may be trained by exposing the model to a large number of emails that have been labeled as either spam or not spam. The model identifies and thus learns the traits and characteristics found in these various emails that are indicative of a spam email or not spam email. Automatically adjusting the

parameters of the model to minimize the times when a spam email is marked as not spam, or vice versa, is the goal of parameter-based training.

21. AI models are generally trained for a specific purpose. However, training a full model is time consuming and, in some cases, has given way to the use of foundational models, which are models that are pre-trained on large, genericized datasets. These foundational models can later be fine-tuned for use in addressing a particular problem or issue by, for example, feeding it a set of additional relevant data and training upon that dataset. By way of example, a foundational AI model for diagnosing medical conditions could be fine-tuned by training it using various medical records. Similarly, a foundational large language model ("LLM") like ChatGPT could be fine-tuned to adjust its conversational tone by training it on sample texts representing a desired communication style. Using a foundational AI model is like getting a shirt and later tailoring it to fit your body, rather than starting with fabric, thread, and a needle.

22. As AI models become more advanced and versatile, they also demand more data and computing power to be properly trained. One of the main challenges in this exploding field of technology is efficiently training these models, which typically is done by configuring variables in a model's learning process. These variables are known as hyperparameters. The hyperparameters may, for example, define the architecture (and thus the size) of the model, as well as, how the model changes based on the training data it digests and that will affect how efficiently and effectively the model is trained. For example, a hyperparameter called the learning rate may be used to control how much a model adjusts its parameters during training. If the learning rate of the model is too low, it will take a long time for the model to reach an acceptable result. On the other hand, if the learning rate is too high, the model may overreact to each training round, bouncing around an optimum result instead of converging on that result. Different hyperparameters

may be used depending on whether a user is training a brand-new model or fine-tuning a foundational model. In either case, though, identifying the right hyperparameters and their values during the model training process improves the model's efficiency and thus the predictions or decisions to be output by the model.

23. Claims of the '120 Patent are directed to technological solutions that address specific challenges grounded in the systematic and automatic tuning of hyperparameters so that efficient and effective AI models can be computer generated. Creating these models is a tremendous concern for modern enterprises, since they are now being used in all manner of human endeavor, from drug discovery, to providing banking efficiencies, etc.

24. The inventor of the '120 Patent understood that while basic models that trained on parameters existed prior to 2007, as the data to be analyzed become more and more voluminous and complex, the use of just parameters in developing a model would not be enough. Rather, the additional use of hyperparameters that could automatically and systematically be optimized by the computing system that was training the model was needed. The inventor also realized that the existing methods for estimating and selecting hyperparameters were unsuitable in many cases. For example, the '120 Patent discloses:

Hyperparameters may be determined using various methods, for example heuristic or statistical optimization methods. However, many heuristic methods are ad hoc methods, while not all statistical optimization methods are suitable for determining hyperparameters.

The inventor thus recognized that a new paradigm needed to be developed. That paradigm was then well before its time invented and is disclosed in the '120 Patent.

25. For example, the '120 Patent explains that a method of determining hyperparameters in a machine learning system is disclosed. The method is iterative, and with each computer implemented iteration, a random or pseudorandom sample of hyperparameter vectors

selected from a set of possible hyperparameter vectors must be considered, updating the estimate of the target hyperparameter vector by selecting for use the best vector in the present and previous iterations. The '120 Patent specification further explains that:

By selecting from the random sample a or the hyperparameter vector producing the best result in the present and any previous iterations, and using said selected hyperparameter vector to update the estimate of the target hyperparameter vector, the properties of the best performing hyperparameters are used to guide the next iteration [of the hyperparameter selection process]. In this way, a method similar to the cross-entropy method [used for selection of parameters] can be used effectively for determining hyperparameters, even when the hyperparameters are not continuous and their effects are not transparent. By using the method according to the present invention, an improved classification [i.e. model] accuracy is achieved.

...

In a preferred embodiment, the method according to the Present invention further comprises the steps of:

selecting from the random sample a further hyperparameter vector
producing the best result in any previous iterations, and
restricting the random sample of hyperparameter vectors by using the
further selected hyperparameter vector.

The step of restricting the random sample preferably involves the use of an interval surrounding the further selected hyperparameter vector which produces the best result in any previous iteration.

...

In a preferred embodiment, E^t is the hyperparameter vector X^t , at iteration t producing the best result $S(X^t_i)$ in the present iteration t and all previous iterations (if any). The random sample X^t_i , has elements X^t_{ij} , while the step of updating the hyperparameters comprises the step of determining the (target) hyperparameter v^t_j , where

$$v^t_j = \frac{\sum_{i=1}^n I\{S(X^t_i) \geq \gamma^t\} W(X^t_i; E^t) X^t_{ij}}{\sum_{i=1}^n I\{S(X^t_i) \geq \gamma^t\} W(X^t_i; E^t)},$$

wherein γ^t is a threshold value, W is a weighting function and $I\{S(X^t_i) \geq \gamma^t\}$ is an indicator function which is equal to 1 if $S(X^t_i)$ greater than or equal to the threshold value γ^t , and else is equal to 0. The weighting function W is preferably given by

$$W(X_i^t; E^t) = 1 - \frac{\sqrt{\sum_{j=1}^m (X_{ij}^t - E_j^t)^2}}{\sqrt{\sum_{j=1}^m (X_{ij}^t)^2} \sqrt{\sum_{j=1}^m (E_j^t)^2}}.$$

Accordingly, the step of updating the hyperparameters may involve a weighting function based upon a distance function, preferably a Euclidean distance function. It is noted that when the Euclidean distance between E^t and X_{ij}^t , is zero, the weighting function W is equal to one.

The method according to the present invention is typically carried out by computer apparatus, for example a general purpose computer system comprising a processor and an associated memory, one part of the memory storing a soft ware program for instructing the processor to carry out the method steps of the present invention, and another part of the memory storing data, said data comprising the hyperparameter values referred to above.

See, e.g., Ex. 1 at 2:1–4:14.

26. The solution to the problems identified in the '120 Patent—as specified and claimed in the '120 Patent—was an advanced departure from the conventional tuning of hyperparameters in AI models in use in 2007 and was then, as it remains today, patent-eligible, highly valuable, novel, and non-obvious technology.

3. K.Mizra's Asserted '120 Patent Claims Are Presumed Valid and Patent-Eligible

27. The '120 Patent is directed to a critical advancement in machine learning, namely, an effective method of estimating hyperparameters given large potential datasets. The sole named inventor of the '120 Patent, Stephan Raaijmakers, is a pioneer in artificial intelligence, holding a Ph.D. in Machine Learning from Tilburg University and a chair on Communicative AI at Leiden University. Dr. Raaijmakers is also the author of the book *Deep Learning for Natural Language Processing*, distributed by Simon & Schuster.

28. K.Mizra asserts that at least, and without limitation, Claim 1 of the '120 Patent has been directly infringed, either literally or under the doctrine of equivalents. K.Mizra reserves the right to assert additional claims of the '120 Patent, including both independent and dependent claims, pursuant to the Court's (and other applicable) rules and procedures and as discovery progresses. These claims are referred to herein as the "Asserted '120 Claims."

29. None of the Asserted '120 Claims are directed to abstract ideas, and each employs inventive concepts and is directed to patent-eligible subject matter. All claims of the '120 Patent are also presumed to be valid and enforceable against Google and others.

30. Indeed, the '120 Patent's specification and claims demonstrate that the need satisfied by the inventions of the Asserted '120 Claims was long-felt in the industry and thus unconventional. The '120 Patent is directed to the use of "hyperparameters" to train machine learning systems. "Hyperparameters" are parameters that can be used to determine, for example, decision criteria to be used in classifying objects. (Ex. 1 at 1:20–23.) Hyperparameters in turn must be determined using particular methods. The '120 Patent specification references two papers discussing a method of estimating parameters called "cross-entropy" and explains why this approach is not suitable for determining hyperparameters in the machine learning context. (*Id.* at 3:44–59.) Rather, the '120 Patent describes an iterative method of determining hyperparameters by drawing a random sample of hyperparameter vectors from a set of possible hyperparameter vectors, selecting from the sample a hyperparameter vector producing the best result, updating the estimate using the selected hyperparameter vector, and repeating the process to determine the best hyperparameters. (*Id.* at 2:1–19.)

31. The '120 Patent thus identifies the conventional method of determining parameters and explains a specific technical improvement, namely:

The present invention is based upon the insight that methods similar to the cross-entropy method can be adapted for hyperparameter estimation by adding a best performance preservation feature. The present invention benefits from the additional insight that the best performance in a particular iteration may be preserved by restricting the sample using the best performance of the previous iterations.

(*Id.* at 6:61–67.) The '120 Patent thus specifically explains a technological solution to improving known methods of machine learning parameter estimation, and the Asserted '120 Claims directed to those improvements are patent eligible. *See, e.g., Aon Re, Inc. v. Zesty.AI, Inc.*, 791 F. Supp. 3d 531, 539 (D. Del. 2025) ("Although the patent does not purport to invent machine learning algorithms, computer hardware, or the general concept of aerial imagery analysis or risk assessment, it does specifically define how these known components must be structured to achieve a supposed technological improvement in accuracy and efficiency to the combination of those components."); *cf. Recentive Analytics, Inc. v. Fox Corp.*, 134 F.4th 1205, 1213 (Fed. Cir. 2025) (holding claims ineligible where "the claims do not delineate steps through which the machine learning technology achieves an improvement").

32. Because the Asserted '120 Claims are directed to a specific improvement in machine learning technology, the claims recite inventions with specific applications or improvements to technologies in the marketplace and cannot be considered abstract or patent ineligible under relevant law.

4. Google's Infringing Vertex AI Studio Product

33. Google's Vertex AI Studio is part of the Google Cloud Platform and is a complete system for building, training, and using AI models. It offers a range of tools and APIs (Application Programming Interfaces) that let users either create and train their own models from scratch, or work with Google's foundation models, like Gemini, Imagen, Codey, and Chirp, and tailor them for specific needs. As is here relevant, Vertex AI provides several options for training all manner

of models. One of the most powerful of these provides a user with complete control over the model development process by allowing them access to powerful hyperparameter tuning and optimization options. These can be implemented using Google's Vertex AI Vizier. It is Google's provision of this computer implemented hyperparameter tuning ability that takes advantage of iterative, randomized/pseudo-randomized hyperparameter tuning techniques, that makes Google's Vertex AI so valuable to its customers. Google's usurpation of the critical '120 Patent's patented technology without K.Mizra's authorization allows Google to provide to its customers highly tuned and effective AI models, generating significant revenues and reputational prestige for Google, all to the injury of K.Mizra.

34. Google has been making, selling, using and offering for sale AI products and services that infringe the '120 Patent in violation of 35 U.S.C. § 271 (collectively, "the Accused Hyperparameter Tuning Instrumentalities"). These Accused Hyperparameter Tuning Instrumentalities include, but are not limited to, Google's Vertex AI system, the sale, offer for sale, use, and/or manufacture in the United States of which constitutes infringement of at least Claim 1 of the '120 Patent, either literally or under the doctrine of equivalents.

B. The '717 Patent

35. K.Mizra is the sole owner by assignment of the '717 Patent with the full and exclusive right to enforce the Patent. K.Mizra is also entitled to sue to collect damages for all past infringement of the '717 Patent.

36. The '717 Patent, titled "Initialization of a Wireless Communication Network," was issued by the USPTO to Inventors Blange and Djurica on March 27, 2012. A true and correct copy of the '717 Patent is attached hereto as Exhibit 2 and incorporated by reference.

37. The '717 Patent is a U.S. National Phase filing of PCT/NL2007/0000001, filed on December 2, 2007, and is entitled to that priority date. The '717 Patent further claims priority to European Patent Application No. EP 05078044, filed December 30, 2005, and is entitled to that priority date.

1. Prior Licensing and Litigation Concerning the '717 Patent

38. The Mesh Networking Patent has previously been licensed to various industry participants.

39. Google is not and has never been a licensee of the '717 Patent nor has it had nor does it have any rights to use technologies covered by the Patent. Google thus has no ownership or other rights (and is entitled to no rights) relating to the '717 Patent.

40. No party licensed to the '717 Patent has used the technology claimed by the '717 Patent in a manner that requires marking under 35 U.S.C. § 287. K.Mizra is therefore not barred from seeking past damages for infringement of the '717 Patent under 35 U.S.C. § 287(a).

2. Problems in 2007 Solved By the '717 Patent

41. The technology described in the '717 Patent was invented by Marinus Johannes Blange and Miodrag Djurica and generally relates to creation of a flexible mesh wireless network.

42. Although wireless communications networks were known in the art prior to the invention, the known networks had several drawbacks. For example, there existed known techniques for routing messages in wireless networks. *See generally* Ex. 2 at 1:13–58. However, as explained in the '717 Patent, these techniques "require[d] transmission of information via sensor stations before the routing topology [of the network] has been defined." *Id.* at 1:59–62. "This complicates the (sensor) stations, because they will have to support different modes of

communication while operating in a low-power mode." *Id.* at 1:62–64. As further explained in the '717 Patent, the existing techniques also resulted in either slow or inefficient networks:

If forwarding of information is only done using the definite routing topology, a central approach is slow (each station has to wait until its upward station has been centrally assigned a route) and a decentral approach is inefficient (a routing topology with an unnecessary amount of branches has been created).

Id. at 1:64–2:2.

43. Another known protocol was a mobile communications protocol which uses two types of networks: a high power network and a low power network. *See generally* Ex. 2 at 2:9–57. However, the '717 Patent notes that "[t]he use of two types of network without central unit makes this system complex and inefficient in bandwidth use." *Id.* at 3:1–2.

44. The solution to these problems—as specified and claimed in the '717 Patent—was an advanced departure from the conventional networking solutions then in use and was then, as it remains today, patent eligible, highly valuable, novel, and non-obvious technology.

3. K.Mizra's Asserted '717 Patent Claims Are Presumed Valid and Patent-Eligible

45. K.Mizra asserts that at least, and without limitation, Claim 11 of the '717 Patent has been directly infringed, either literally or under the doctrine of equivalents by Google. K.Mizra reserves the right to assert additional claims of the '717 Patent, including both independent and dependent claims, pursuant to the Court's (and other applicable) rules and procedures and as discovery progresses. These claims are referred to herein as the "Asserted '717 Claims."

46. None of the Asserted '717 Claims are directed to abstract ideas, and each employs inventive concepts and are directed to patent-eligible subject matter. All claims of the '717 Patent are also presumed to be valid and enforceable against Google and others.

47. Indeed, the '717 Patent's specification and claims demonstrate that the need satisfied by the inventions of the Asserted '717 Claims was long-felt in the industry and thus

unconventional. As one example, the '717 Patent explains that stations within wireless networks were required to transmit information before routing topology had been defined, *i.e.* when plural routes were still possible. The '717 Patent explains that this created complications because wireless stations would have to support different modes of communication while operating in low-power states, which was difficult to achieve. The '717 Patent further explains that wireless links between nodes were hindered by barriers and that using the same frequency for upward and downward link routes required synchronization, which led to further complications by requiring wireless stations to support synchronization and powerful radio frequency signals, again at low-power states. The '717 Patent thus provides "a wireless network wherein routes for communication can be determined without the need to forward information via stations for which the routes have not yet been defined," a method of operating such a network, and "stations for flexible use in wireless networks." (Ex. 2 at 3:6–15.)

48. The specification (including the provisions quoted above), the figures (including those included below), and the text related to the figures further illustrate the innovative network architecture captured by the Asserted '717 Claims. These figures include the following:

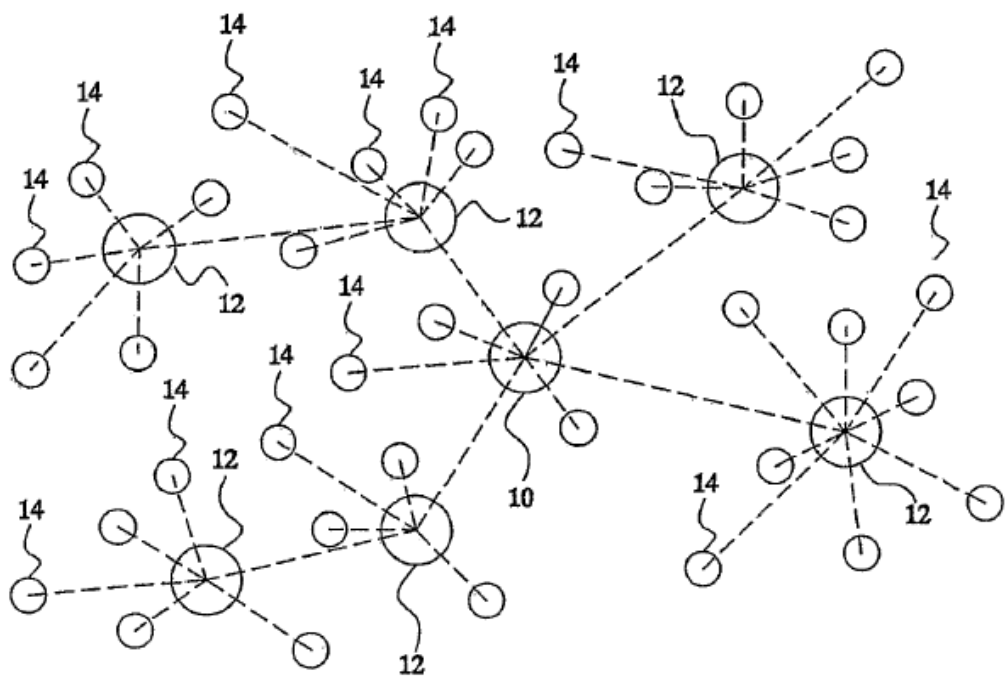


Fig.1

(See *id.* at Fig. 1.)

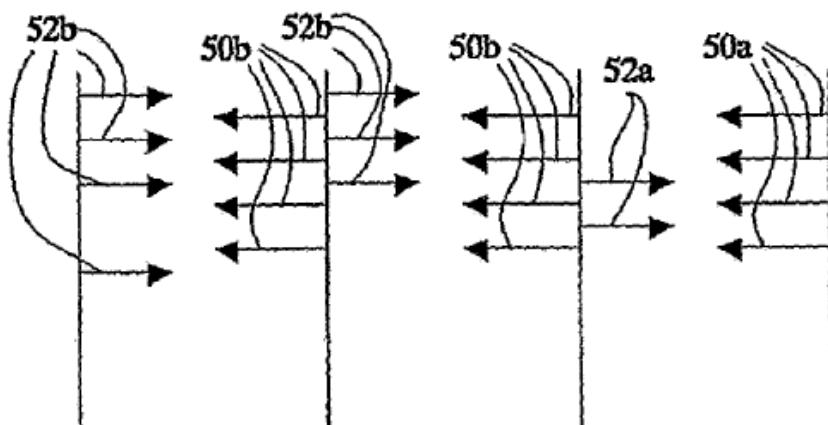


Fig.5a

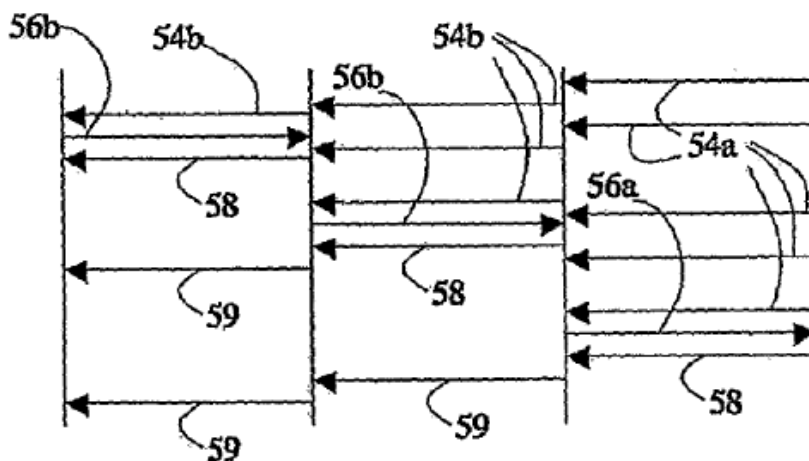


Fig.5b

(See *id.* at Fig. 5.)

49. The foregoing figures, which illustrate a wireless network and signaling protocol, demonstrate that the inventions of the Asserted '717 Claims focus on specific, highly efficient ways for a station to join a wireless network, thereby improving the operation and performance of the wireless network, as a whole. In particular, an unassociated station transmits an association request message and switches to an associated state upon reception of an association grant from

an association unit in response to one of its request messages. The association grant establishes an operating route associated with the newly associated station for exchanging operating messages with the association unit. Thereafter, the station can transmit association grants in response to received association request messages from additional stations seeking to join the network. This improves the efficiency and operability of the wireless network by facilitating the addition of new stations to the network. Thus, the Asserted '717 Claims are eligible as a matter of law for patent protection under step one of *Alice*.

50. All actions and steps recited in the Asserted '717 Claims, including the acts of transmitting association request messages, transmitting association grant messages, establishing operating routes and exchanging operating messages, and enabling stations to transmit association grant messages, require the involvement of various hardware components running dedicated software both before, during, and after the association of a station. Said another way, a claim directed to allowing a wireless network to automatically and dynamically associate a new station with the network and facilitate the addition of other stations is not simply adding a generic computer component to a fundamentally human process. Rather, it is an "improvement in the functioning of a" networked system that simply cannot be considered directed to an abstract concept. *Enfish LLC v. Microsoft Corp.*, 822 F.3d 1327, 1339 (Fed. Cir. 2016).

51. As the specification confirms, the improvement captured by the Asserted '717 Claims is not simply associating various devices into a wireless network, but is instead a specific network architecture whereby a particular unit operates to receive access requests and provide access grants, as well as providing operation information so that newly added wireless stations can transmit access grants to additional stations. Specifically, the specification of the '717 Patent discloses that to improve the efficiency of adding stations to a wireless network, a station seeking

to join the network operates initially in a non-associated state transmitting access requests. (*E.g.*, Ex. 2 at 8:25–32.) Subsequently, the station receives an access grant message and switches to an associated state. (*Id.* at 9:3–13.) In this state, the station establishes a dedicated communication route with the association unit or with a second station acting as an intermediary between the newly added station and the association unit, thereby ensuring efficient communication within the wireless network. (*Id.*) The newly added station can thereafter relay association grant messages to additional new stations, further enhancing the scope and efficiency of the wireless network. (*Id.* at 9:41–53.)

52. The '717 Patent specification further emphasizes that, once stations are added to the network via association grant messages, they have "well defined routes" to communicate within the network. (*Id.* at 9:63–10:6.) Thus, rather than requiring specific access routes in order to join the network initially, which creates needless complexity and power usage, the '717 Patent teaches that well defined routes are established after the station has been added to the network via an efficient method.

53. The Asserted '717 Claims are thus directed to a machine-implemented process for efficiently adding a new station to a wireless network and configuring that station to facilitate the addition of further stations to the network, improving the overall operation of the network itself. As such, the Asserted '717 Claims recite inventions with specific applications or improvements to technologies in the marketplace and cannot be considered abstract or patent ineligible under relevant law.

4. Google's Infringing Nest Devices

54. Google Nest is a line of products for home networking, allowing a user to control various Google and third-party smart devices throughout the Google Home app. On information and belief, Google Nest was previously known as Google Home.

55. Google offers a wide variety of home networking devices under the Google Nest brand. Many of these devices use a networking protocol called Thread.

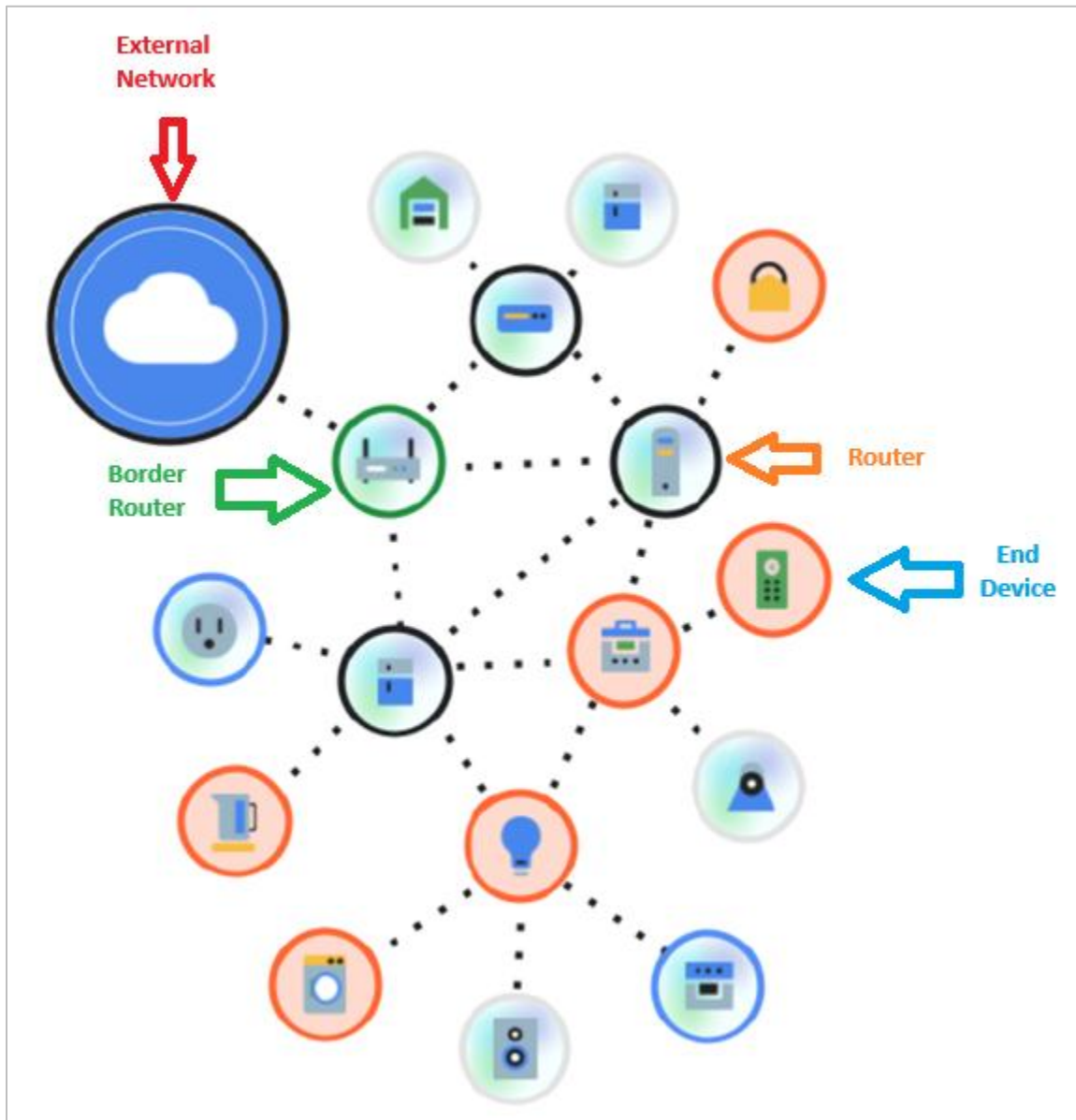
56. Thread is a networking protocol designed for low-power Internet of Things devices operating in an IEEE 802.11.15.4 wireless mesh network. It may be used alongside other networks, such as a Wi-Fi network, in a user's home. Thread was developed by the Thread group, which Google founded with other member companies in 2014.

57. Thread is built into many new smart devices such as smart bulbs, locks, sensors, and the like. Thread is used to create a mesh network in which each new smart device connects to the next nearest smart device, giving users more networking coverage and making the entire network more robust. If one device in the network fails, another can step in to maintain the signal, allowing the network to heal itself rapidly, effectively and efficiently.

58. A device in a Thread network may be either a router, which may forward data packets for other network devices and provides commissioning for devices trying to join the Thread network, or an end device, which communicates with a single router and does not forward packets for other network devices. An end device may convert itself into a router when a new device joins the network.

59. A special case of a router is a border router, which can forward information between a Thread network and a non-Thread network. For example, a border router may allow a user to control the devices on a Thread network through the Wi-Fi network via a border router.

60. An exemplary Thread network including a border router, routers, and end devices is shown below:



"Thread Play Services APIs," (available at <https://developers.home.google.com/matter/thread> (accessed March 2026 and incorporated by reference) (annotated)).

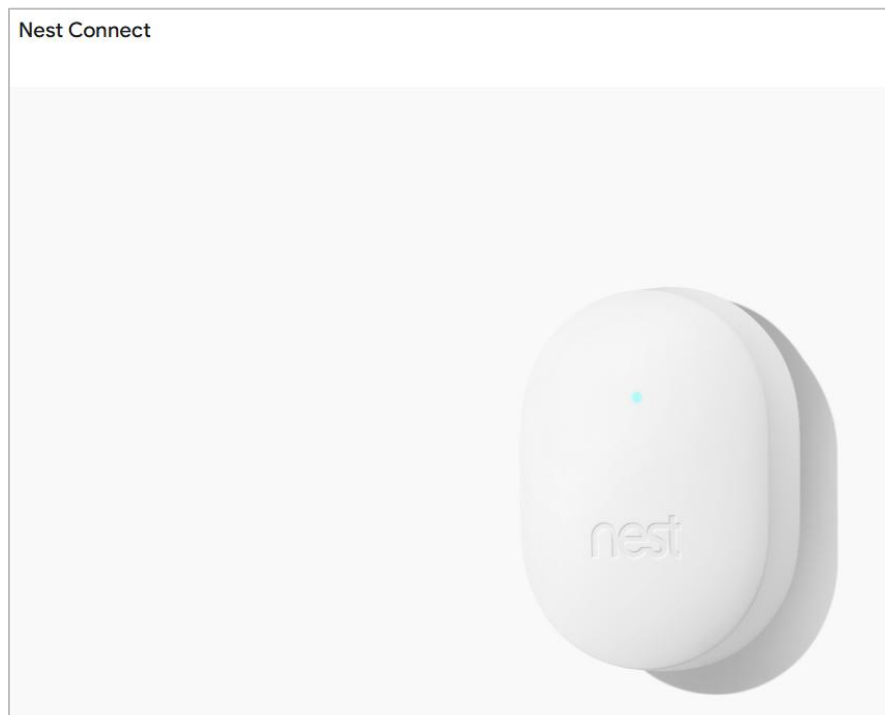
61. Google offers a wide range of products that use the Thread networking protocol. For example, Google offers several hubs for Google home that include a built-in Thread router, including the Nest Hub, Nest Hub Max, Nest Wifi Pro, and Google TV Streamer.

Google devices like the Nest WiFi, Google Nest Hub Max and Google Nest Hub (2nd gen) have Thread radios built-into them and act as Thread Border Routers.

Id.; see also "Connect Thread smart devices with Google TV Streamer (4K)," (available at <https://support.google.com/chromecast/answer/15178609> (accessed March 2026 and incorporated by reference)).

62. In addition, Google hub products such as the Google TV Streamer, Nest Hub (2nd Gen), Hub Nest Max Fuchsia, Nest Wi-Fi Point, and Nest Wi-Fi Router are certified as Built on Thread by the Thread Group.

63. As another example, Google offers a routing device called the Nest Connect.



Nest Connect – Range Extender for Detect & Guard – Google Store," (available at https://store.google.com/product/nest_connect (accessed March 2026 and incorporated by reference)). The Nest Connect includes a Thread radio that allows communication between a Thread border router and a peripheral device such as the Nest x Yale Lock.

In an emailed statement, Grant Erickson, president of the Thread Group said, "Nest Detect relies on Thread's low power architecture to last for long periods of time on one battery. Nest Connect is a Thread router, which seamlessly extends both the reach and connectivity of the Thread network. Nest Guard provides seamless IP-based routing between Thread, Wi-Fi, and cellular, and their interoperability with the Yale Linus Lock is the posterchild of Thread's benefits in action."

"Nest debuts Nest Secure home system and has a Thread router called Nest Connect," (available at <https://staceyoniot.com/nest-debuts-nest-secure-home-system-and-has-a-thread-router-called-nest-connect/> (accessed March 2026 and incorporated by reference)).

64. Yet another example, Google offers various peripheral devices that include a Thread radio such as the Nest x Yale Lock.



"Nest x Yale Lock – Key-Free Smart Deadbolt – Google Store," (available at https://store.google.com/product/nest_x_yale_lock (accessed March 2026 and incorporated by reference)).

65. Additional peripheral devices offered by Google that include a Thread radio include the Google Nest Learning Thermostat and the Google Nest Protect and CO alarm.

66. Google has been making, selling, using and offering for sale products including Thread radios that infringe the '717 Patent in violation of 35 U.S.C. § 271 (collectively, "the Accused Mesh Networking Instrumentalities"). These Accused Mesh Networking Instrumentalities include, but are not limited to, Google's Nest home networking products identified above, the sale, offer for sale, use, and/or manufacture in the United States of which constitutes infringement of at least Claim 11 of the Accused Mesh Networking Patent, either literally or under the doctrine of equivalents.

67. K.Mizra expects that discovery will reveal additional products that include Thread radios and infringe the '717 Patent, which may include additional Google Nest products and/or products that are not marketed under the Google Nest brand. For example, there are reports that at least some Google Pixel 9 devices may include a Thread radio. *See, e.g.*, "Pixel 9 series arrives at the FCC, includes Thread," (available at <https://9to5google.com/2024/07/12/pixel-9-fcc/> (accessed March 2026 and incorporated by reference)).

C. Google's Knowledge of the Asserted Patents

68. Google has been aware of K.Mizra's patent portfolio, including the Asserted Patents, based on discussions between the Parties in 2022 and early 2023.

69. Google has had actual knowledge of the Asserted Patents and its infringement of the Asserted Patents since at least May 23, 2025. On May 23, 2025, K.Mizra sent Google a

proposed amended complaint in *K.Mizra LLC v. Google LLC*, Case No. 1:25-cv-00236-ADA (W.D. Tex.). The proposed amended complaint included allegations of infringement of the Asserted Patents.

V. FIRST CLAIM FOR RELIEF
(Patent Infringement Under 35 U.S.C. § 271 of the '120 Patent)

70. K.Mizra incorporates paragraphs 1 through 69 as though fully set forth herein.

71. The '120 Patent includes 16 claims.

72. Google has directly infringed one or more claims of the '120 Patent by making, importing, using, offering for sale, and/or selling the Accused Hyperparameter Tuning Instrumentalities, in violation of 35 U.S.C. § 271(a).

73. Based on publicly available information, the Accused Hyperparameter Tuning Instrumentalities satisfy every limitation of at least Claim 1 of the '120 Patent.

74. Claim 1 of the '120 Patent states:

[preamble] A method of determining hyperparameters of a classifier in a machine learning system by iteratively producing an estimate of a target hyperparameter vector, each iteration comprising the steps of:

[A] drawing a random sample of hyperparameter vectors from a set of possible hyperparameter vectors,

[B] updating the estimate of the target hyperparameter vector by using the random sample, and

[C] selecting, from the random sample of hyperparameter vectors, a hyperparameter vector producing a best result in the present and any previous iterations, and wherein the step of updating the estimate of the target hyperparameter vector uses said hyperparameter vector producing the best result.

75. Regarding the preamble of Claim 1, and to the extent the preamble is determined to be limiting, the Accused Hyperparameter Tuning Instrumentalities provide the features described in the preamble, which recites "A method of determining hyperparameters." For

example, Google touts that its Vertex AI Platform is a "fully-managed, unified AI development platform for building and using generative AI."; "Custom training gives you complete control over the training process, including using your preferred ML framework, writing your own training code, and choosing hyperparameter tuning options."; and "Hyperparameter tuning jobs run multiple trials of your training application using different hyperparameter values."



Vertex AI Platform

Innovate faster with enterprise-ready AI, enhanced by Gemini models

Vertex AI is a fully-managed, unified AI development platform for building and using generative AI. Access and utilize [Vertex AI Studio](#), [Agent Builder](#), and 200+ foundation models.

- **Custom training gives you complete control over the training process, including using your preferred ML framework, writing your own training code, and choosing hyperparameter tuning options.**

"Innovate faster with enterprise-ready AI, enhanced by Gemini models," (available at <https://cloud.google.com/vertex-ai?hl=en> (accessed March 2026 and incorporated by reference)).

Hyperparameter optimization



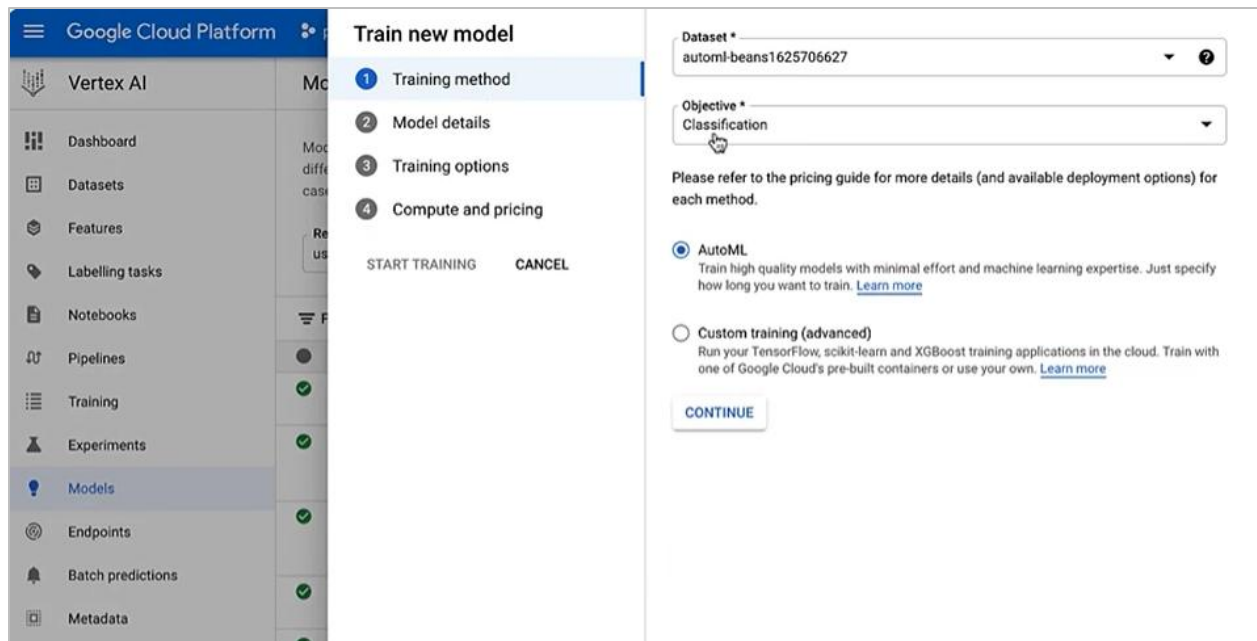
Hyperparameter tuning jobs run multiple trials of your training application using different hyperparameter values. You specify a range of values to test, and Vertex AI discovers the optimal values for your model within that range.

"Vertex AI custom training overview," (available at <https://cloud.google.com/vertex-ai/docs/training/overview> (accessed March 2026 and incorporated by reference)). Accordingly, and to the extent the preamble of Claim 1 is deemed limiting, the Accused Hyperparameter Tuning Instrumentalities meet the limitation.

76. The preamble also includes "of a classifier in a machine learning system".

Vertex AI is a machine learning (ML) platform that lets you train and deploy ML models and AI applications. Vertex AI combines data engineering, data science, and ML engineering workflows, which lets teams collaborate using a common toolset.

"Vertex AI documentation," (available at <https://docs.cloud.google.com/vertex-ai/docs> (accessed March 2026 and incorporated by reference)).



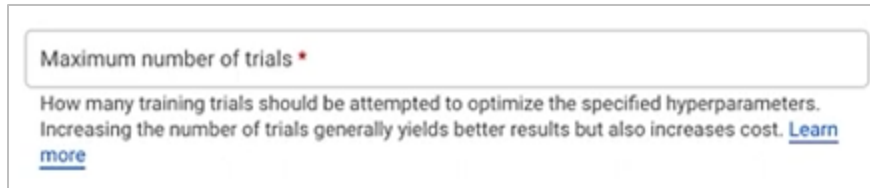
See screenshot above of page at https://www.youtube.com/watch?v=8hZ_cBwNOss at 6:30 (cropped) (accessed March 2026 and incorporated by reference). Thus, and again only to the extent that this portion of the preamble of Claim 1 is deemed limiting, the Accused Hyperparameter Tuning Instrumentalities meet the limitation.

77. The preamble further includes "by iteratively producing an estimate of a target hyperparameter vector":

How hyperparameter tuning works

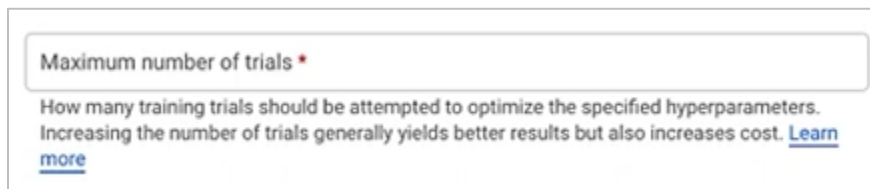
Hyperparameter tuning works by running multiple *trials* of your training application with values for your chosen hyperparameters, set within limits you specify. Vertex AI keeps track of the results of each trial and makes adjustments for subsequent trials. When the job is finished, you can get a summary of all the trials along with the most effective configuration of values according to the criteria you specify.

"Overview of hyperparameter tuning," (available at <https://docs.cloud.google.com/vertex-ai/docs/training/hyperparameter-tuning-overview> (accessed March 2026 and incorporated by reference)).



See screenshot above of page at https://www.youtube.com/watch?v=8hZ_cBwNOss at 7:13 (cropped) (accessed March 2026 and incorporated by reference). Accordingly, and to the extent this portion of the preamble of Claim 1 is deemed limiting, the Accused Hyperparameter Tuning Instrumentalities meet the limitation.

78. The preamble further also includes "each iteration comprising the steps of:"



Id. Again, then, to the extent this portion of the preamble of Claim 1 is deemed limiting, the Accused Hyperparameter Tuning Instrumentalities meet the limitation.

79. Accordingly, the Accused Hyperparameter Tuning Instrumentalities meet limitation of the preamble of Claim 1.

80. Limitations A and B of Claim 1 require:

[A] drawing a random sample of hyperparameter vectors from a set of possible hyperparameter vectors,

[B] updating the estimate of the target hyperparameter vector by using the random sample, and

Google has disclosed:

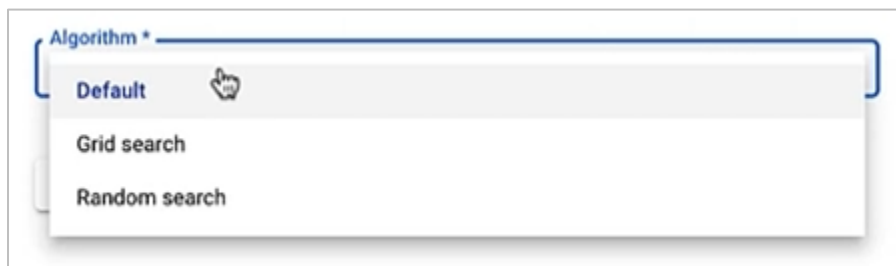
Search algorithms

You can specify a search algorithm in the `StudySpec` object. If you don't specify an algorithm, your job uses the default Vertex AI algorithm. The default algorithm applies Bayesian optimization to arrive at the optimal solution with a more effective search over the parameter space.

Available values:

- `ALGORITHM_UNSPECIFIED` : Same as not specifying an algorithm. Vertex AI chooses the best search algorithm between Gaussian process bandits, linear combination search, or their variants.
- `GRID_SEARCH` : A grid search within the feasible space. This option is particularly useful if you want to specify a quantity of trials that is greater than the number of points in the feasible space. In such cases, if you don't specify a grid search, the Vertex AI default algorithm may generate duplicate suggestions. To use grid search, all parameters must be of type `INTEGER` , `CATEGORICAL` , or `DISCRETE` .
- `RANDOM_SEARCH` : A random search within the feasible space.

"Overview of hyperparameter tuning," (available at <https://docs.cloud.google.com/vertex-ai/docs/training/hyperparameter-tuning-overview> (accessed March 2026 and incorporated by reference)).



See screenshot above of page at https://www.youtube.com/watch?v=8hZ_cBwNOss at 8:43 (cropped) (accessed March 2026 and incorporated by reference).

81. Limitation A requires "drawing a random sample of hyperparameter vectors from a set of possible hyperparameter vectors" and the Google system meets this requirement:

```

StudySpec studySpec =
    StudySpec.newBuilder()
        .addMetrics(metric)
        .addParameters(parameter)
        .setAlgorithm(StudySpec.Algorithm.RANDOM_SEARCH)
        .build();
CustomJobSpec trialJobSpec =
    CustomJobSpec.newBuilder().addWorkerPoolSpecs(workerPoolSpec).build();
// hyperparameter_tuning_job
HyperparameterTuningJob hyperparameterTuningJob =
    HyperparameterTuningJob.newBuilder()
        .setDisplayName(displayName)
        .setMaxTrialCount(4)
        .setParallelTrialCount(2)
        .setStudySpec(studySpec)
        .setTrialJobSpec(trialJobSpec)
        .build();

```

"Create a hyperparameter tuning job for python package," (available at <https://docs.cloud.google.com/vertex-ai/docs/samples/aipatform-create-hyperparameter-tuning-job-python-package-sample?hl=en> (accessed March 2026 and incorporated by reference)).

82. Accordingly, the Accused Hyperparameter Tuning Instrumentalities meet Limitation A of Claim 1.

83. Limitation B requires "updating the estimate of the target hyperparameter vector by using the random sample". The use of random search meets this limitation:

Random search: set up a grid of hyperparameter values and select *random* combinations to train the model and score. The number of search iterations is set based on time/resources.

Random search is surprisingly efficient compared to grid search. Although grid search will find the optimal value of hyperparameters (assuming they are in your grid) eventually, random search will usually find a "close-enough" value in far fewer iterations. [This great paper explains why this is so](#): grid search spends too much time evaluating unpromising regions of the hyperparameter search space because it has to evaluate every single combination in the grid. Random search in contrast, does a better job of exploring the search space and therefore can usually find a good combination of hyperparameters in far fewer iterations.

"Intro to Model Tuning: Grid and Random Search," (available at <https://www.kaggle.com/code/willkoehrsen/intro-to-model-tuning-grid-and-random-search> (accessed March 2026 and incorporated by reference)).

Randomized Search for Hyperparameter Estimation

Randomized search is another method for hyperparameter optimization that can be more efficient than grid search in some cases. With randomized search, instead of specifying a list of values for each hyperparameter, you specify a distribution for each hyperparameter. The randomized search algorithm will then sample values for each hyperparameter from its corresponding distribution and train a model using the sampled values. This process is repeated a specified number of times, and the optimal values for the hyperparameters are chosen based on the performance of the models.

"Comparing Randomized Search and Grid Search for Hyperparameter Estimation in Scikit Learn," (available at <https://www.geeksforgeeks.org/comparing-randomized-search-and-grid-search-for-hyperparameter-estimation-in-scikit-learn/> (accessed March 2026 and incorporated by reference)).

84. Accordingly, the Accused Hyperparameter Tuning Instrumentalities also meet Limitation B of Claim 1.

85. Limitation C requires "selecting, from the random sample of hyperparameter vectors, a hyperparameter vector producing a best result in the present and any previous iterations, and wherein the step of updating the estimate of the target hyperparameter vector uses said hyperparameter vector producing the best result." Google describes:

How hyperparameter tuning works

Hyperparameter tuning works by running multiple *trials* of your training application with values for your chosen hyperparameters, set within limits you specify. Vertex AI keeps track of the results of each trial and makes adjustments for subsequent trials. When the job is finished, you can get a summary of all the trials along with the most effective configuration of values according to the criteria you specify.

"Overview of hyperparameter tuning," (available at <https://docs.cloud.google.com/vertex-ai/docs/training/hyperparameter-tuning-overview> (accessed March 2026 and incorporated by reference)).

86. Accordingly, the Accused Hyperparameter Tuning Instrumentalities meet Limitation C of Claim 1.

87. Additionally, and/or alternatively, Google has indirectly infringed and continues to indirectly infringe one or more claims of the '120 Patent, in violation of 35 U.S.C. § 271(b) by actively inducing users of the Accused Hyperparameter Tuning Instrumentalities to directly infringe one or more claims of the '120 Patent. For example, (a) Google had actual knowledge of or was willfully blind to the existence of the '120 Patent and its infringement thereof no later than its receipt of the proposed amended complaint alleging infringement of the '120 Patent in May 2025, and (b) Google intentionally causes, urges, or encourages users of the Accused Hyperparameter Tuning Instrumentalities to take action that, when taken, directly infringes one or more claims of the '120 Patent. Google's encouragement is accomplished by promoting, advertising, and instructing customers and potential customers to use the Accused Hyperparameter Tuning Instrumentalities, including infringing uses thereof. Google knows (based at least on the proposed amended complaint provided in May 2025 and/or after reading this Complaint) that its actions will induce users of the Accused Hyperparameter Tuning Instrumentalities to directly

infringe one or more claims of the '120 Patent, and users thereof directly infringe one or more claims of the '120 Patent. For instance, at a minimum, Google has supplied and continues to supply the Accused Hyperparameter Tuning Instrumentalities to customers while knowing that use thereof will infringe one or more claims of the '120 Patent.

88. Due to Google's knowledge of the '120 Patent and its infringement thereof no later than May 2025, Google's infringement of the '120 Patent has been and continues to be willful since no later than that date.

89. Google's acts of infringement have occurred within this District and elsewhere throughout the United States.

90. As a result of Google's infringing conduct, K.Mizra has suffered damages. Google is liable to K.Mizra in an amount that adequately compensates K.Mizra for Google's infringement in an amount that is no less than a to-be-calculated fully paid-up, lump-sum, reasonable royalty, together with interest and costs as fixed by this Court under 25 U.S.C. § 284.

VI. SECOND CLAIM FOR RELIEF
(Patent Infringement Under 35 U.S.C. § 271 of the '717 Patent)

91. K.Mizra incorporates paragraphs 1 through 90 as though fully set forth herein.

92. The '717 Patent includes 20 claims.

93. Google has directly infringed one or more claims of the '717 Patent by making, importing, using, offering for sale, and/or selling the Accused Mesh Networking Instrumentalities, in violation of 35 U.S.C. § 271(a).

94. Based on publicly available information, the Accused Mesh Networking Instrumentalities satisfy every limitation of at least Claim 11 of the '717 Patent.

95. Open Thread is an open-source implementation of the Thread networking protocol released by Google. A description of Open Thread is provided at [www.openthread.io](https://openthread.io). On information and belief, openthread.io is owned and operated by Google.

96. On information and belief, Open Thread is an implementation of the Thread networking technology used in Google Nest products.



OpenThread released by Google is an open-source implementation of Thread[®]. Google has released OpenThread to make the networking technology used in Google Nest products more broadly available to developers, in order to accelerate the development of products for the connected home and commercial buildings.

"OpenThread" (available at <https://openthread.io/> (accessed March 2026 and incorporated by reference)).

97. Claim 11 of the '717 Patent states:

[preamble] A station for use in a wireless communication network, the station comprising:

[A] a transmitter arrangement, a receiver arrangement, and a processor circuit coupled to the transmitter arrangement and the receiver arrangement,

[B] the processor configured to start up in a not-associated state and to cause the transmitter arrangement to transmit association request messages in the not-associated state,

[C] the processor circuit being configured to switch to an associated state upon reception at the receiver arrangement of an association grant in response to one of the association request messages,

[D] the processor circuit being configured to establish an associated route for communication with an association unit during subsequent communication of operating messages to an association unit,

[E] the associated route including a source of the association grant and, if the source of the association grant is not the association unit, the operating route between the source of the association grant and the association unit previously established for the source of the association grant in response to association request messages and/or association grants for intermediate stations,

[F] the processor circuit being configured to cause the transmitter arrangement to initiate and transmit association grants in response to association request messages, but only after switching to the associated state.

98. Regarding the preamble of Claim 11, and to the extent the preamble is determined to be limiting, the Accused Mesh Networking Instrumentalities provide the features described in the preamble, which recites "A station for use in a wireless communication network." For example, Google devices including the Nest WiFi, the Nest Hub Max, and the Nest Hub (2nd Gen) are stations configured for use in a wireless network implementing the Thread networking protocol.



A more reliable smart home network with Thread.

Thread is the latest technology for keeping all your smart home devices connected. Nest Hub Max and Nest Hub (2nd gen) have a built-in Thread border router to connect your Thread mesh network of smart devices to your Wi-Fi network. Your Thread-enabled network will be faster, use less energy, and reach more places throughout and around your home.

"Nest Hub Smart Displays for your Home – Google Store," (available at https://store.google.com/category/nest_hubs_displays (accessed March 2026 and incorporated by reference)).

Thread devices join users' existing home networks through a Thread Border Router. Just like a Wi-Fi router can bridge Wi-Fi and Ethernet devices into a single network, a Thread Border Router allows Thread devices to become part of users' networks.

Google devices like the Nest WiFi, Google Nest Hub Max and Google Nest Hub (2nd gen) have Thread radios built-into them and act as Thread Border Routers.

"Thread Play Services APIs," (available at <https://developers.home.google.com/matter/thread> (accessed March 2026 and incorporated by reference)).

99. Similarly, the Nest Connect is a station configured for use in a Thread wireless network.

In an emailed statement, Grant Erickson, president of the Thread Group said, "Nest Detect relies on Thread's low power architecture to last for long periods of time on one battery. Nest Connect is a Thread router, which seamlessly extends both the reach and connectivity of the Thread network. Nest Guard provides seamless IP-based routing between Thread, Wi-Fi, and cellular, and their interoperability with the Yale Linus Lock is the posterchild of Thread's benefits in action."

"Nest debuts Nest Secure home system and has a Thread router called Nest Connect," (available at <https://staceyoniot.com/nest-debuts-nest-secure-home-system-and-has-a-thread-router-called-nest-connect/> (accessed March 2026 and incorporated by reference)).

100. Accordingly, and to the extent the preamble of Claim 11 is deemed limiting, the Accused Mesh Networking Instrumentalities meet the limitation.

101. Limitation A of Claim 11 of the '717 Patent requires a transmitter arrangement, a receiver arrangement, and a processor circuit coupled to the transmitter arrangement and the receiver arrangement.

102. The Accused Mesh Networking Instrumentalities include a Thread radio with a transmitter arrangement and a receiver arrangement, as well as a processor circuit coupled to the transmitter arrangement and receiver arrangement that processes transmissions and receptions. For example, Google's documents confirm that each of the Nest WiFi, the Google Hub Max, and the Google Nest Hub (2nd Gen) include Thread radios.

Thread has the following key benefits:

- IPv6 based: Thread devices can join the same network as your other devices, and talk directly to each other and the cloud.
- Low-powered mesh: Built for IoT, Thread supports battery operated devices, with a mesh that delivers range and reliability.
- Speed: Low overhead, local connectivity, and mesh makes Thread devices extremely responsive.

Thread is developed in the Thread Group, which Google founded with other member companies in 2014.

Thread devices join users' existing home networks through a Thread Border Router. Just like a Wi-Fi router can bridge Wi-Fi and Ethernet devices into a single network, a Thread Border Router allows Thread devices to become part of users' networks.

Google devices like the Nest WiFi, Google Nest Hub Max and Google Nest Hub (2nd gen) have Thread radios built-into them and act as Thread Border Routers.

"Thread Play Services APIs," (available at <https://developers.home.google.com/matter/thread> (accessed March 2026 and incorporated by reference)). Similarly, a statement from the Thread Group (of which Google is a member) confirms that the Nest Connect is a Thread router.

In an emailed statement, Grant Erickson, president of the Thread Group said, "Nest Detect relies on Thread's low power architecture to last for long periods of time on one battery. Nest Connect is a Thread router, which seamlessly extends both the reach and connectivity of the Thread network. Nest Guard provides seamless IP-based routing between Thread, Wi-Fi, and cellular, and their interoperability with the Yale Linus Lock is the posterchild of Thread's benefits in action."

"Nest debuts Nest Secure home system and has a Thread router called Nest Connect," (available at <https://staceyoniot.com/nest-debuts-nest-secure-home-system-and-has-a-thread-router-called-nest-connect/> (accessed March 2026 and incorporated by reference)). Indeed, in order to communicate with other devices, each Thread device (including the Accused Mesh Networking Instrumentalities) must necessarily include a transmitter arrangement (to transmit messages from

the Thread device to other devices in the Thread network) and a receiver arrangement (to receive communications, including responses to its transmitted messages, from other devices in the Thread network). Thread devices, including the Accused Mesh Networking Instrumentalities, must also include a processor coupled to both the transmitter arrangement and the receiver arrangement to, among other things, initiate transmissions, respond to messages sent by other devices in the Thread network, and process received messages. The operations of the processor are described in further detail in connection with Limitations B through F.

103. Accordingly, the Accused Mesh Instrumentalities meet Limitation A of Claim 11.

104. Limitation B of Claim 11 of the '717 Patent requires the processor is configured to start up in a not-associated state and to cause the transmitter arrangement to transmit association request messages in the not-associated state.

105. The Accused Mesh Networking Instrumentalities include this feature. For example, Thread devices, such as the Accused Mesh Networking Devices, have a Detached state when the Thread device has not yet attached to a Thread network.

Note: The Detached state is observed when Thread is enabled but the device is not attached to a Thread network. The Disabled state is observed when Thread is disabled on the device.

"Developing with OpenThread APIs, (available at <https://openthread.io/codelabs/openthread-apis#1> (accessed March 2026 and incorporated by reference)).

106. In order to attach to a Thread network, a Thread device such as the Accused Mesh Networking Instrumentalities goes through the Attach process.

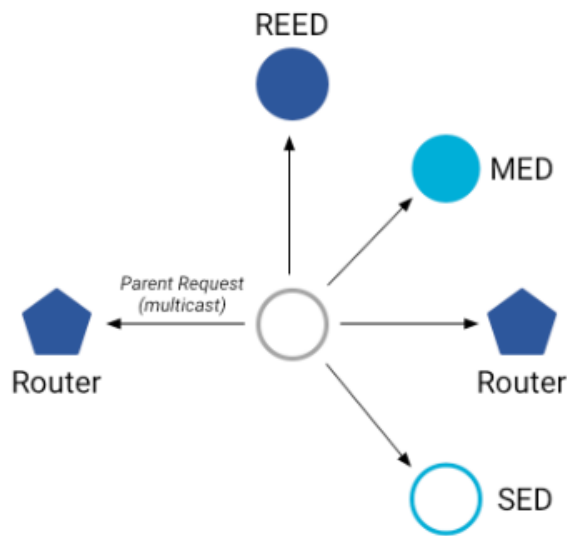
Join an existing network

If the device elects to join an existing network, it configures its Channel, PAN ID, XPAN ID, and Network Name to match that of the target network via Thread Commissioning, then goes through the MLE Attach process to attach as a Child (End Device). This process is used for Child-Parent links.

"Network Discovery and Formation," (available at <https://openthread.io/guides/thread-primer/network-discovery> (accessed March 2026 and incorporated by reference)). As the first step in this process, an attaching device sends a Parent Request message to find neighboring devices that are part of the Thread network.

1. Parent Request

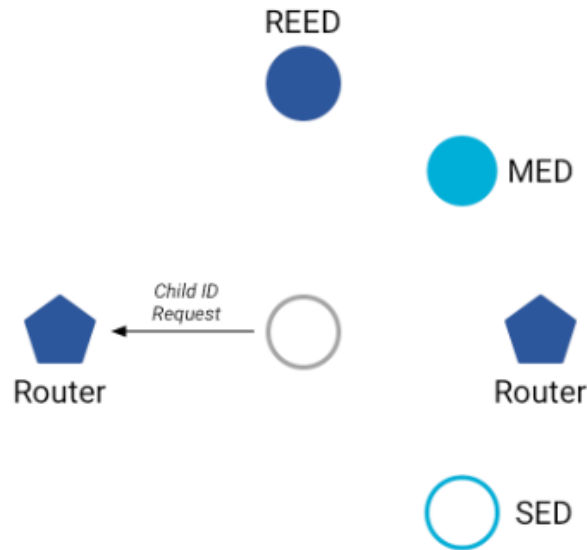
A Parent Request is a multicast request from the attaching device that is used to discover neighboring Routers and Router Eligible End Devices (REEDs) in the target network.



Id. Upon receiving a response, the Thread device sends a message called the Child ID Request to a particular device in the Thread network for the purpose of establishing a link with that device to join the network.

3. Child ID Request

A Child ID Request is a unicast request from the attaching device (Child) that is sent to the Router or REED (Parent) for the purpose of establishing a Child-Parent link. If the request is sent to a REED, it **upgrades itself to a Router** before accepting the request.



Id. The attaching device may also be referred to as the Child, and the router with which the attaching device is seeking to establish a link may be referred to as the Parent.

107. Accordingly, the Accused Mesh Networking Instrumentalities meet Limitation B of Claim 11.

108. Limitation C of Claim 11 of the '717 Patent requires the processor circuit being configured to switch to an associated state upon reception at the receiver arrangement of an association grant in response to one of the association request messages.

109. Thread devices, such as the Accused Mesh Networking Instrumentalities, become attached to a Thread network through the MLE Attach process. As described above in connection with Limitation B, the attaching device begins the process by sending a Parent Request and then exchanges further messages in order to establish a link with a device in the existing Thread network.

Join an existing network

If the device elects to join an existing network, it configures its Channel, PAN ID, XPAN ID, and Network Name to match that of the target network via Thread Commissioning, then goes through the MLE Attach process to attach as a Child (End Device). This process is used for Child-Parent links.



Key Point: Every device, router-capable or not, initially attaches to a Thread network as a Child (End Device).

1. The Child sends a multicast **Parent Request** to all neighboring Routers and REEDs in the target network.
2. All neighboring Routers and REEDs (if the Parent Request Scan Mask includes REEDs) send **Parent Responses** with information about themselves.
3. The Child chooses a Parent device and sends a **Child ID Request** to it.
4. The Parent sends a **Child ID Response** to confirm link establishment.

Id. When the link has been established, the Thread device has attached to the network and thus is in an attached state. The establishment of the link between the attaching device and the Thread device in the existing Thread network is confirmed by the Child ID Response.

4. Child ID Response

A Child ID Response is a unicast response from the Parent that is sent to the Child to confirm that a Child-Parent link has been established.

Id. Notably, Thread devices such as the Accused Mesh Networking Instrumentalities may exchange information about how long they have been in certain states or roles, suggesting that these devices track their current state/role and when each state/role is entered.

<code>mChildRole</code>	<code>uint16_t</code> Number of times device entered child role.
<code>mChildTime</code>	<code>uint64_t</code> Milliseconds device has been in child role.
<code>mDetachedRole</code>	<code>uint16_t</code> Number of times device entered detached role.
<code>mDetachedTime</code>	<code>uint64_t</code> Milliseconds device has been in detached role.

"otNetworkDiagMleCounters," (available at <https://openthread.io/reference/struct/ot-network-diag-mle-counters> (accessed March 2026 and incorporated by reference)).

110. Accordingly, the Accused Mesh Networking Instrumentalities meet Limitation C of Claim 11.

111. Limitation D of Claim 11 of the Accused Mesh Networking Instrumentalities requires the processor circuit being configured to establish an associated route for communication with an association unit during subsequent communication of operating messages to an association unit.

112. Each of the Accused Mesh Networking Instrumentalities is configured to operate in a Thread network. A Thread network includes a device designated as the Leader.

Thread Leader

The Thread Leader is a Router that is responsible for managing the set of Routers in a Thread network. It is dynamically self-elected for fault tolerance, and aggregates and distributes network-wide configuration information.

"Node Roles and Types," (available at <https://openthread.io/guides/thread-primer/node-roles-and-types> (accessed March 2026 and incorporated by reference)). The processors of the Accused Mesh

Networking Devices are configured to store information about the route between the attaching device and the Leader as further discussed below in connection with Limitation E of Claim 11.

113. Accordingly, the Accused Mesh Networking Instrumentalities meet Limitation D of Claim 11.

114. Limitation E of Claim 11 requires the associated route includes a source of the association grant and, if the source of the association grant is not the association unit, the operating route between the source of the association grant and the association unit previously established for the source of the association grant in response to association request messages and/or association grants for intermediate stations.

115. Upon joining a Thread network, Thread devices such as the Accused Mesh Networking Devices receive a Child ID Response which provides information about the route between the Leader and the Thread device. For example, the Child ID Response includes the Source Address (i.e., the address for the Parent Thread device), the Leader RLOC (i.e., the address for the Leader of the Thread network), Network Data such as "more-specific routes," and Route information.

Child ID Response Message Contents	
Source Address	Parent's RLOC16
Address16	Child's RLOC16
Leader Data	Information about the Parent's Leader (RLOC, Partition ID, Partition weight)
Network Data	Information about the Thread network (on-mesh prefixes, address autoconfiguration, more-specific routes)
Route (REED only)	Route propagation
Timeout	Inactivity duration before the Parent removes the Child
Address Registration (MEDs and SEDs only)	Confirm registered addresses

"Network Discovery and Formation," (available at <https://openthread.io/guides/thread-primer/network-discovery> (accessed March 2026 and incorporated by reference)). K.Mizra further expects that discovery, including review of source code, will provide further information and confirm this understanding.

116. Accordingly, the Accused Mesh Networking Instrumentalities meet Limitation D of Claim 11.

117. Limitation E of Claim 11 of the '717 Patent requires the processor circuit being configured to cause the transmitter arrangement to initiate and transmit association grants in response to association request messages, but only after switching to the associated state.

118. Thread devices such as the Accused Mesh Networking Instrumentalities can send Parent Response and Child ID Response communications once attached to the Thread network, but will not send such responses if the device is disconnected from the network. For example, upon joining a Thread network, Router Eligible End Devices (or REEDs) can upgrade to become routers.

Upgrade to a Router

After attaching to a Thread network, the Child device may elect to become a Router. Before initiating the MLE Link Request process, the Child sends an Address Solicit message to the Leader, asking for a Router ID. If the Leader accepts, it responds with a Router ID and the Child upgrades itself to a Router.

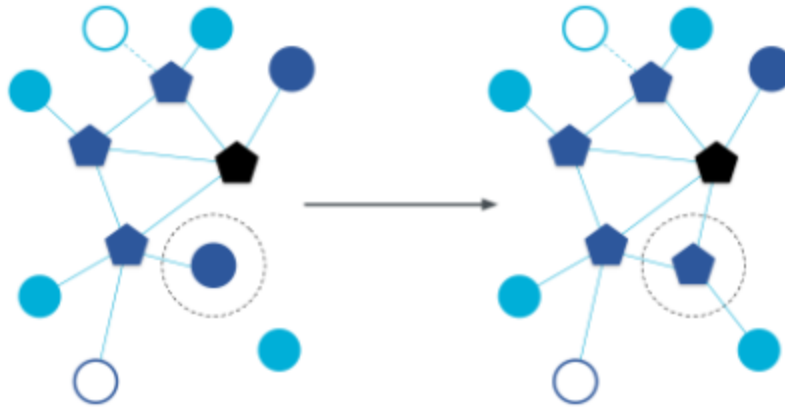
The MLE Link Request process is then used to establish bi-directional Router-Router links with neighboring Routers.

1. The new Router sends a multicast [Link Request](#) to neighboring Routers.
2. Routers respond with [Link Accept and Request](#) messages.
3. The new Router responds to each Router with a unicast [Link Accept](#) to establish the Router-Router link.

"Router Selection," (available at <https://openthread.io/guides/thread-primer/router-selection> (accessed March 2026 and incorporated by reference)).

Upgrading and downgrading

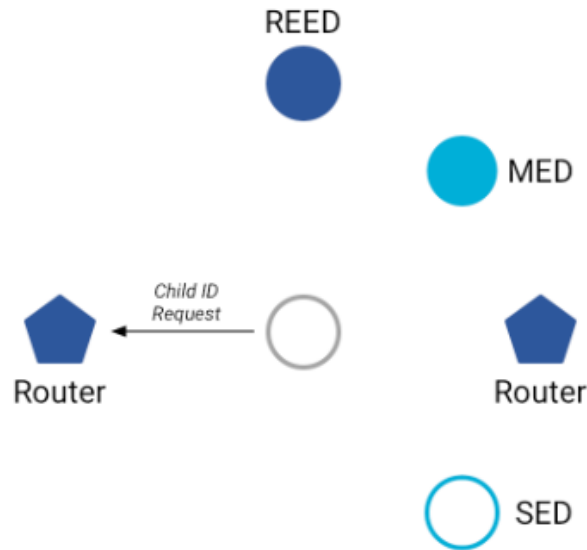
When a REED is the only node in reach of a new End Device wishing to join the Thread network, it can upgrade itself and operate as a Router:



"Node Roles and Types," (available at <https://openthread.io/guides/thread-primer/node-roles-and-types> (accessed March 2026 and incorporated by reference)). Upon upgrading to a router, the Thread device is capable of accepting a request from an attaching device to join the Thread network.

3. Child ID Request

A Child ID Request is a unicast request from the attaching device (Child) that is sent to the Router or REED (Parent) for the purpose of establishing a Child-Parent link. If the request is sent to a REED, it **upgrades itself to a Router** before accepting the request.



"Network Discovery and Formation," (available at <https://openthread.io/guides/thread-primer/network-discovery> (accessed March 2026 and incorporated by reference)).

119. Accordingly, the Accused Mesh Networking Instrumentalities meet limitation of the Limitation E of Claim 11.

120. Additionally, and/or alternatively, Google has indirectly infringed and continues to indirectly infringe one or more claims of the Mesh Networking Patent, in violation of 35 U.S.C. § 271(b) by actively inducing users of the Accused Mesh Networking Instrumentalities to directly infringe one or more claims of the Mesh Networking Patent. For example, (a) Google had actual knowledge of or was willfully blind to the existence of the Mesh Networking Patent and its infringement thereof no later than its receipt of the proposed amended complaint alleging infringement of the Mesh Networking Patent in May 2025, and (b) Google intentionally causes, urges, or encourages users of the Accused Mesh Networking Instrumentalities to take action that,

when taken, directly infringes one or more claims of the Mesh Networking Patent. Google's encouragement is accomplished by promoting, advertising, and instructing customers and potential customers to use the Accused Mesh Networking Instrumentalities, including infringing uses thereof. Google knows (based at least on the proposed amended complaint provided in May 2025 and/or after reading this Complaint) that its actions will induce users of the Accused Mesh Networking Instrumentalities to directly infringe one or more claims of the Mesh Networking Patent, and users thereof directly infringe one or more claims of the Mesh Networking Patent. For instance, at a minimum, Google has supplied and continues to supply the Accused Mesh Networking Instrumentalities to customers while knowing that use thereof will infringe one or more claims of the Mesh Networking Patent.

121. Due to Google's knowledge of the Mesh Networking Patent and its infringement thereof no later than May 2025, Google's infringement of the Mesh Networking Patent has been and continues to be willful since no later than that date.

122. Google's acts of infringement have occurred within this District and elsewhere throughout the United States.

123. As a result of Google's infringing conduct, K.Mizra has suffered damages. Google is liable to K.Mizra in an amount that adequately compensates K.Mizra for Google's infringement in an amount that is no less than a to-be-calculated fully paid-up, lump-sum, reasonable royalty, together with interest and costs as fixed by this Court under 25 U.S.C. § 284.

VII. REQUEST FOR RELIEF

WHEREFORE, K.Mizra respectfully requests the Court find in its favor and against Google, and that the Court grant K.Mizra at least the following relief:

A. Judgment that Google has infringed, literally and/or under the doctrine of equivalents, one or more claims of the Asserted Patents;

B. Awarding damages to K.Mizra in an amount to be proven at trial and in the form of a fully paid-up, lump sum, reasonable royalty that considers and runs through expiration of all the Asserted Patents;

C. Awarding enhanced damages, as appropriate, under 35 U.S.C. § 284;

D. Awarding K.Mizra's costs (including internal and external costs and disbursements) and declaring this an exceptional case and awarding K.Mizra its attorneys' fees in accordance with 35 U.S.C. § 285;

E. Pre-judgment and post-judgment interest at the maximum rate permitted by law on the damages caused by reason of Google's infringing activities and other conduct complained of herein; and

F. Awarding such other and further relief as this Court deems just and proper under the circumstances.

VIII. DEMAND FOR JURY TRIAL

Pursuant to Fed. R. Civ. P. 38(b), K.Mizra hereby demands a trial by jury on all issues so triable.

Dated: March 27, 2026

Respectfully submitted,

By: /s/ Michael C. Smith

Michael C. Smith
Texas Bar No. 18650410
michael.smith@solidcounsel.com
Scheef & Stone, LLP
113 E. Austin Street
Marshall, TX 75670
(903) 938-8900

Robert R. Brunelli*

CO State Bar No. 20070
rbrunelli@sheridanross.com
Matthew C. Holohan*
CO State Bar No. 40996
mholohan@sheridanross.com
Bart A. Starr*
CO State Bar No. 50446
bstarr@sheridanross.com
Brian S. Boerman*
CO State Bar No. 50834
bboerman@sheridanross.com
Tristan D. Lewis*
CO State Bar No. 60968
tlewis@sheridanross.com
SHERIDAN ROSS P.C.
1560 Broadway, Suite 1200
Denver, CO 80202
Telephone: 303-863-9700
Facsimile: 303-863-0223
litigation@sheridanross.com

Of Counsel:

Claire A. Henry
Texas Bar No. 24053063
Miller Fair Henry PLLC
1507 Bill Owens Parkway
Longview, TX 75604
Telephone: (903) 757-6400
Facsimile: (903) 757-2323
claire@millerfairhenry.com

**To be admitted pro hac vice
Attorneys for Plaintiff K.Mizra LLC*